IoTアプリケーション検証のための エッジコンピューティングテストベッド

山中 広明(NICT)

スマートIoT推進フォーラム技術戦略検討部会
第１１回テストベッド分科会
2021/09/27

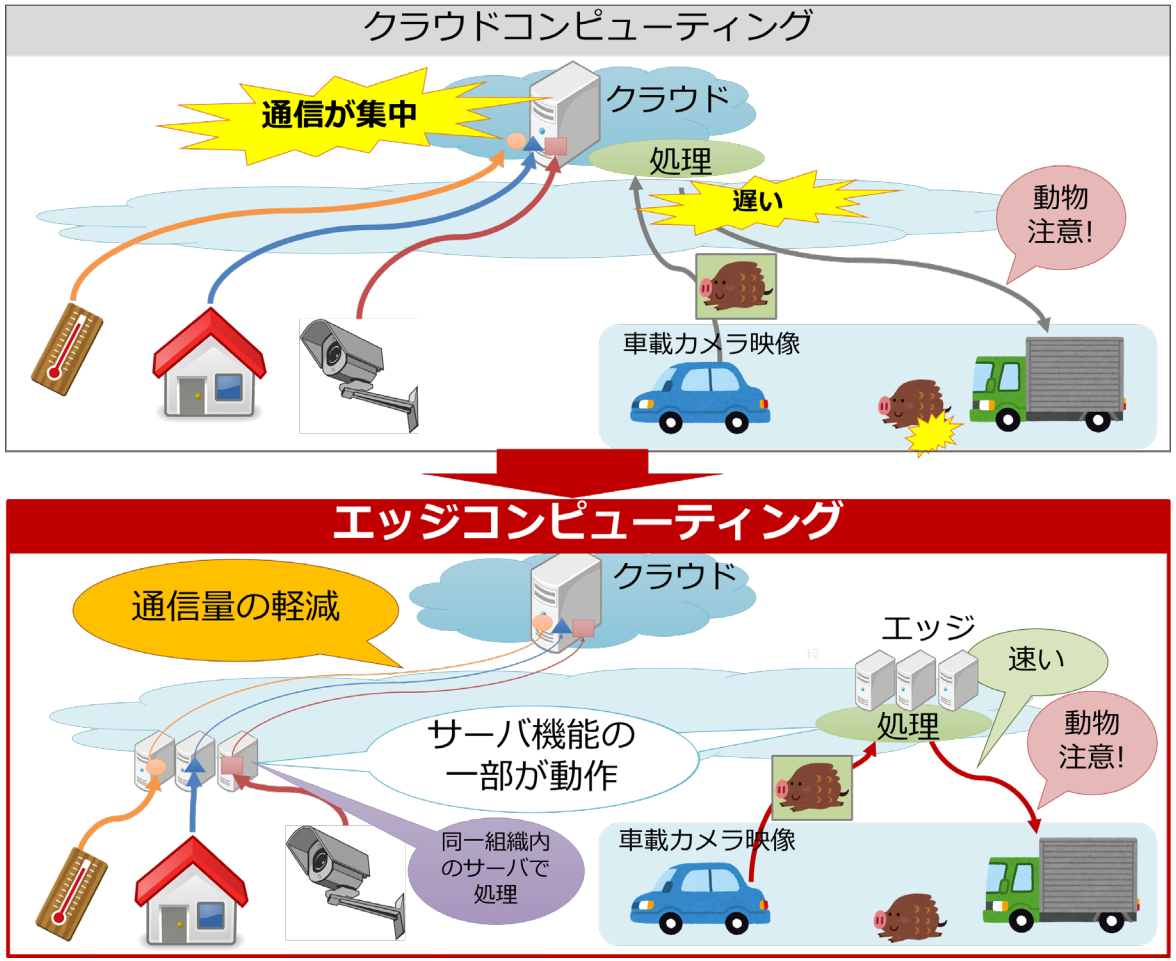
エッジコンピューティング

○エンドユーザデバイス近隣のエッジクラウド内サーバを利用

◇ネットワークの帯域利用抑制

◇高速な応答

◇データ流通地域制限



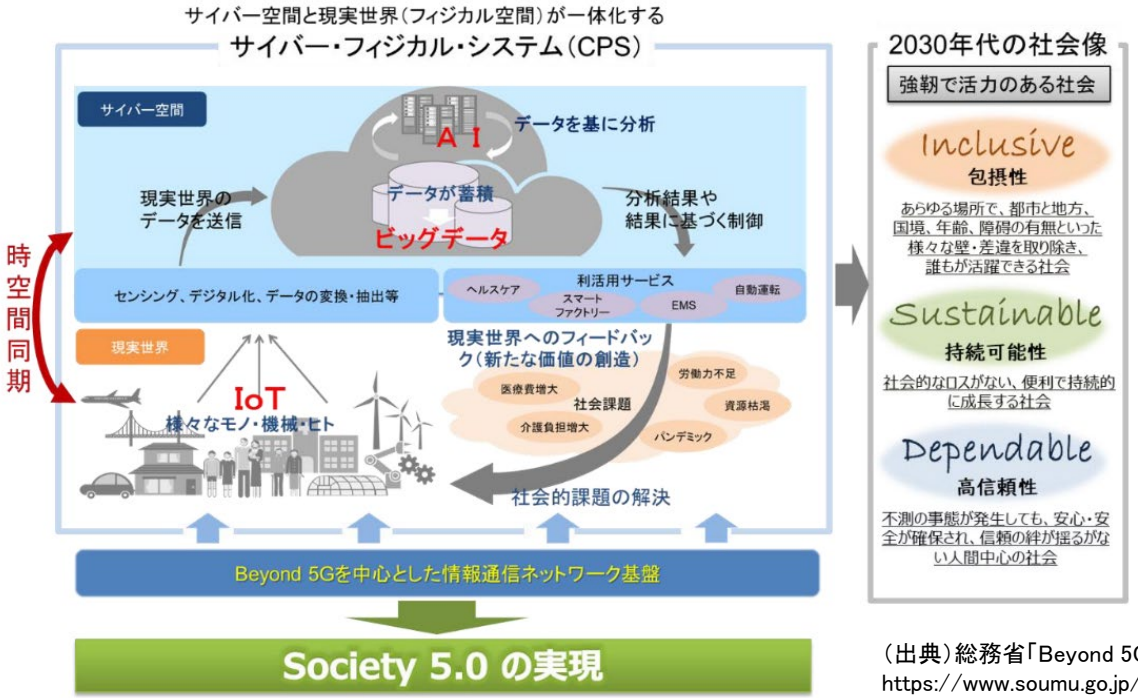
サイバーフィジカルシステム

○サイバーフィジカルシステムの普及が期待

◇膨大なセンサデータをAIが処理，アクチュエータにフィードバック

◇サイバーフィジカルシステムのインフラとしてエッジコンピューティングが期待

（図表１：サイバー・フィジカル・システムの進展と 2030 年代の社会像）



（出典）総務省「Beyond 5G推進戦略懇談会　提言」（2020）
https://www.soumu.go.jp/main_content/000696612.pdf

○エッジコンピューティングが使える検証環境（テストベッド）により，アプリケーションサービスのスムーズな実展開をサポート

◇IoTアプリケーション開発者がテストベッドでアプリケーションプログラムの動作確認

○テストベッド環境構築：膨大な作業を伴う

◇サーバインストール

◇ネットワーク接続

○テストベッドサービスとして提供

◇容易な検証のため，Dockerコンテナに対応

●サーバプログラムを軽量にパッケージング，高速に展開

テストベッドサービス実現の課題

○アプリケーションの応答遅延の条件に応じたサーバへのDockerコンテナ配置

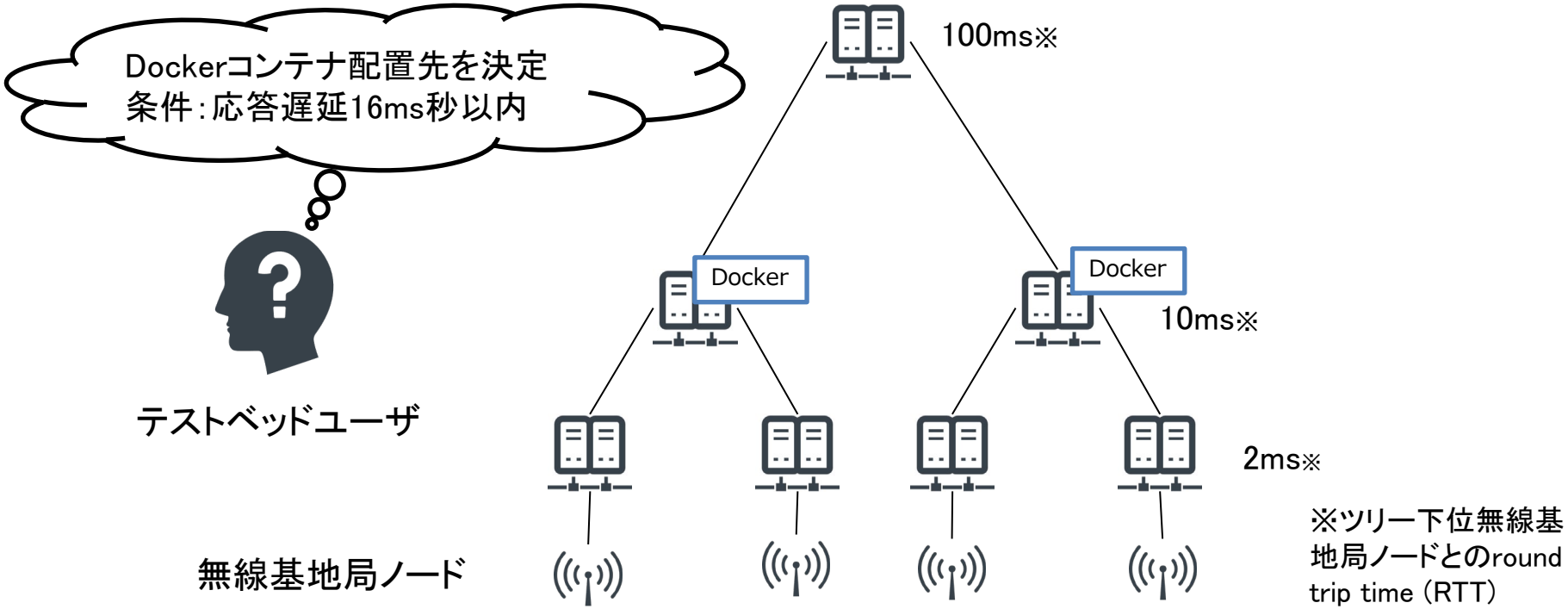
◇応答遅延例 [1]

- AR：16ms
- リモートコントロールロボット：2ms

○既存システム（Kubernetes等）ではテストベッドユーザに複雑な操作が必要

◇ネットワークトポロジや伝搬遅延の確認

◇Dockerコンテナ配置スケジューラの変更



[1] ITU-T, “The Tactile Internet.” Aug. 2014. Available: https://www.itu.int/oth/T2301000023/en

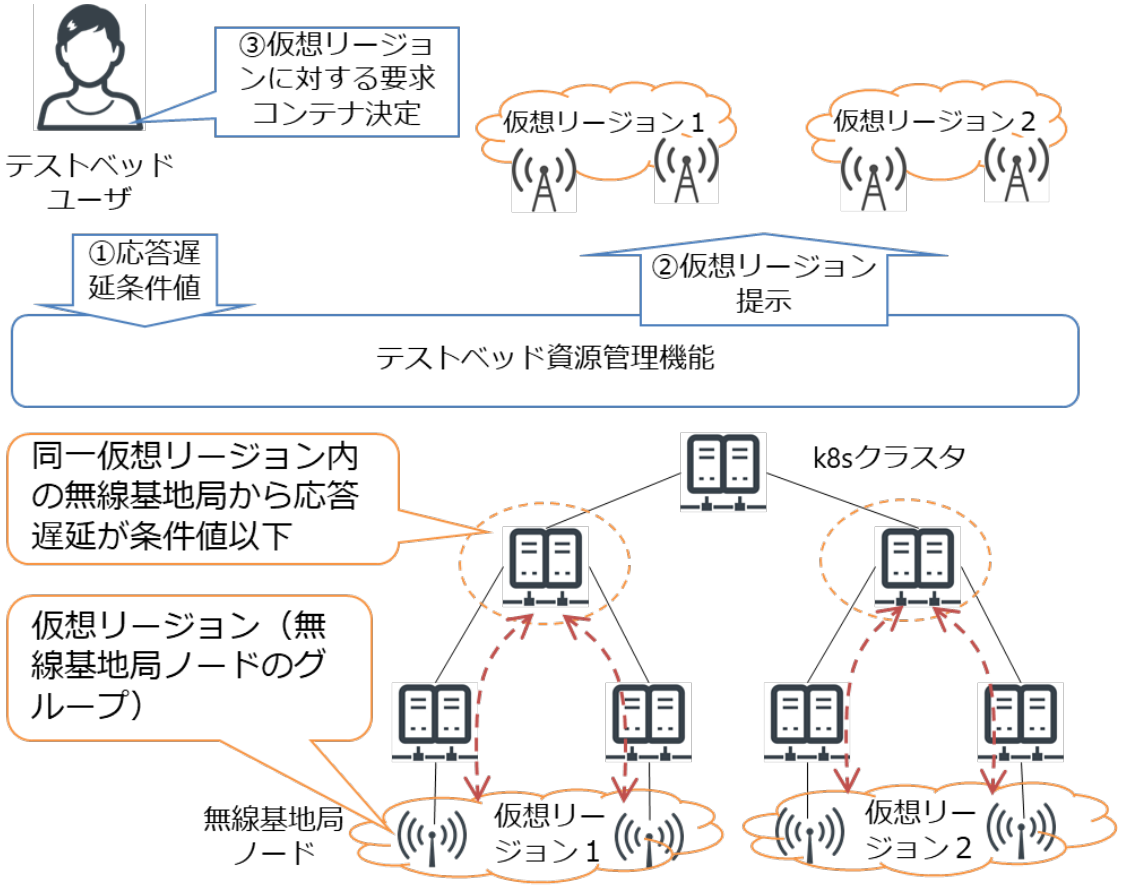
提案：仮想リージョンを適用したユーザ作業の軽減

○「仮想リージョン」による資源要求

◇応答遅延を満たすコンテナの一括配置が可能な無線基地局ノード群（仮想リージョン）をユーザに提示

◇仮想リージョンに対するコンテナ配置要求→応答遅延を満たす資源配置

○ユーザによるDockerコンテナ配置において，ネットワークトポロジ，伝搬遅延の確認が不要



システム利用例

```
edgeiaas@uadmtest:~/etbcmd/src$ python create_vr.py 10
('POST: ', 'http://192.168.11.30:5500/providers/p1/virtual_regions')
('body: ', {'latency': 10.0})
200
{
 "virtual_regions": [
  {
   "aps": [
    {
     "name": "AP79"
    }
   ],
   "latency": 10.0,
   "name": "vr318"
  },
  {
   "aps": [
    {
     "name": "AP80"
    }
   ],
   "latency": 10.0,
   "name": "vr155"
  },
```

①応答遅延条件(RTT ms)提示

②仮想リージョン提示：{<無線基地局ノードリスト>，<応答遅延(RTT)>，<仮想リージョン名>}をJSON形式で返答

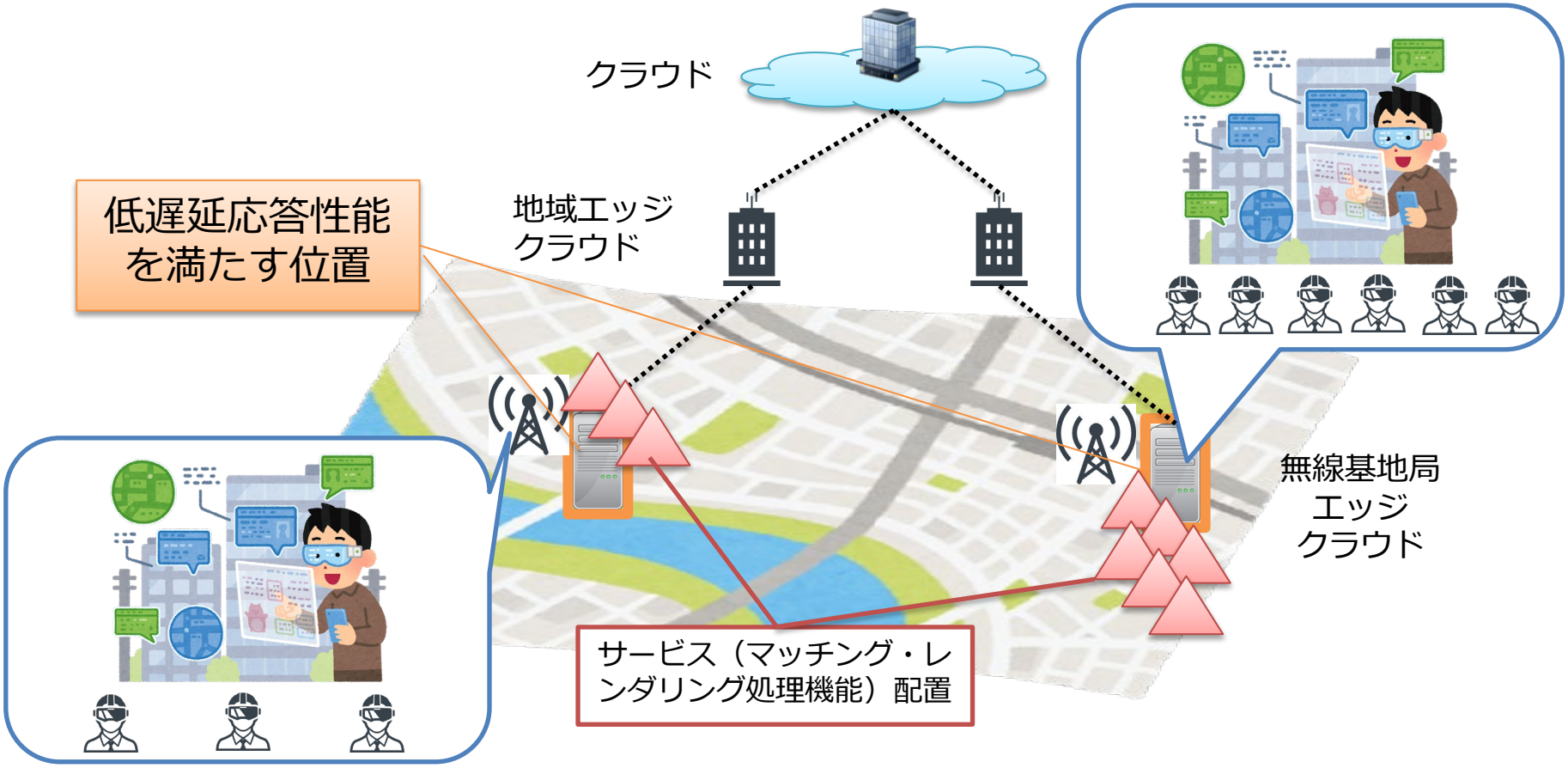
③仮想リージョンに対するコンテナ要求

```
edgeiaas@uadmtest:~/etbcmd/src$ python create_con.py vr318 192.168.11.103/myimage
('POST: ', 'http://192.168.11.30:5500/providers/p1/virtual_regions/vr318/containers')
200
 * Serving Flask app "create_con" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://192.168.11.103:5500/ (Press CTRL+C to quit)
{
  "container_info": [
    {
      "edge_cloud": "ec8",
      "exp_wan_ip": "172.19.135.64",
      "name": "p1-vr318-mycontainer-1",
      "pod_ip": "172.18.156.197",
      "pod_port": 80,
      "port": 10001,
      "protocol": "TCP",
      "ssh_port": 20001,
      "ssh_wan_ip": "192.168.11.64",
      "status": "Created",
      "vr": "vr318"
    }
  ]
}
192.168.11.30 - - [19/Nov/2020 11:38:20] "PUT /containers/p1-vr318-mycontainer HTTP/1.1" 200 -
```

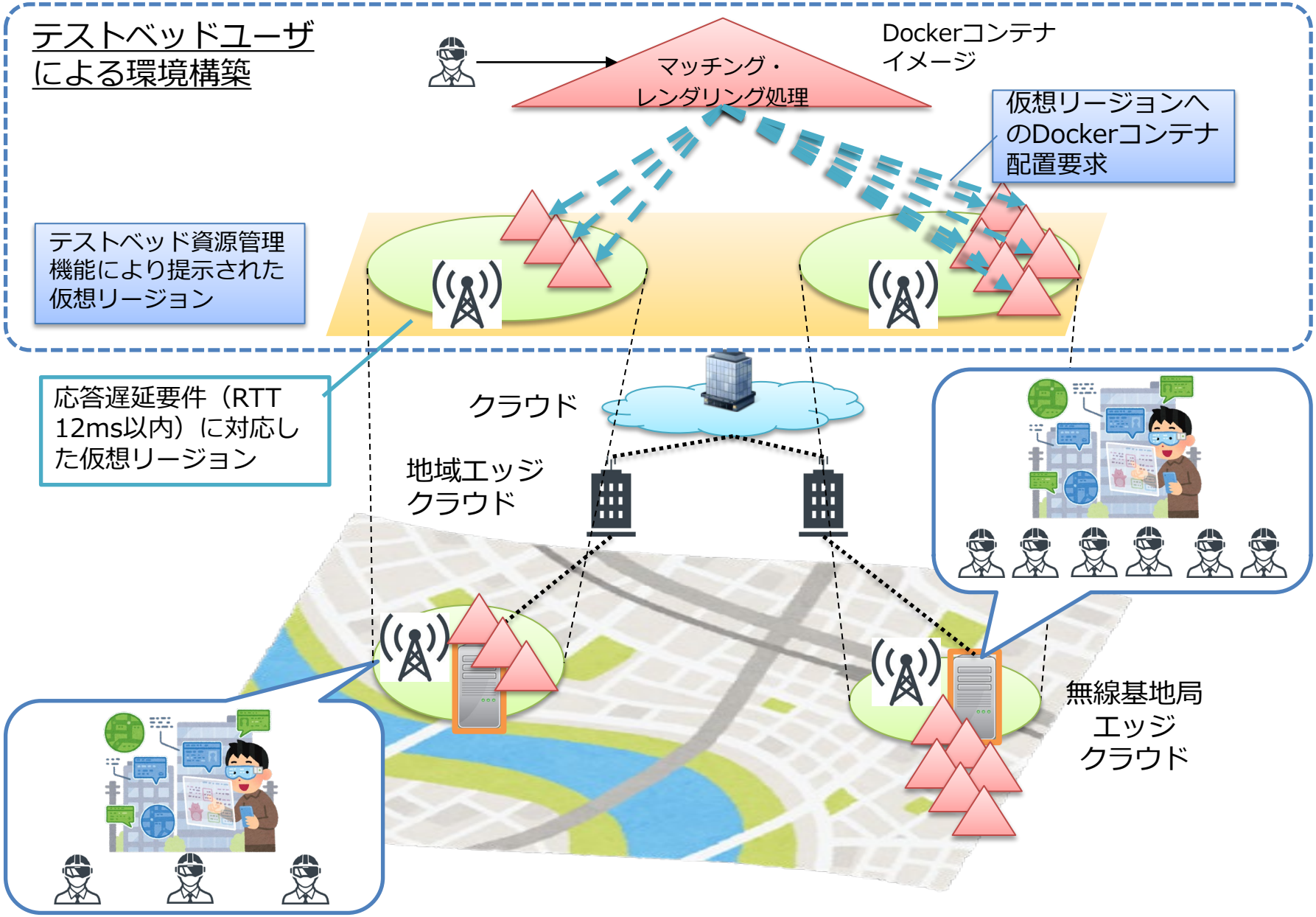
作成されたDockerコンテナのIPアドレス，サービスポート番号等をJSON形式で返答

※①～③の作業をコマンド１つで実行するスクリプトを用意，より簡易な利用も可能

○低遅延応答性能を満たす位置（例：RTT 12ms以内）のエッジクラウドにDockerコンテナ配置

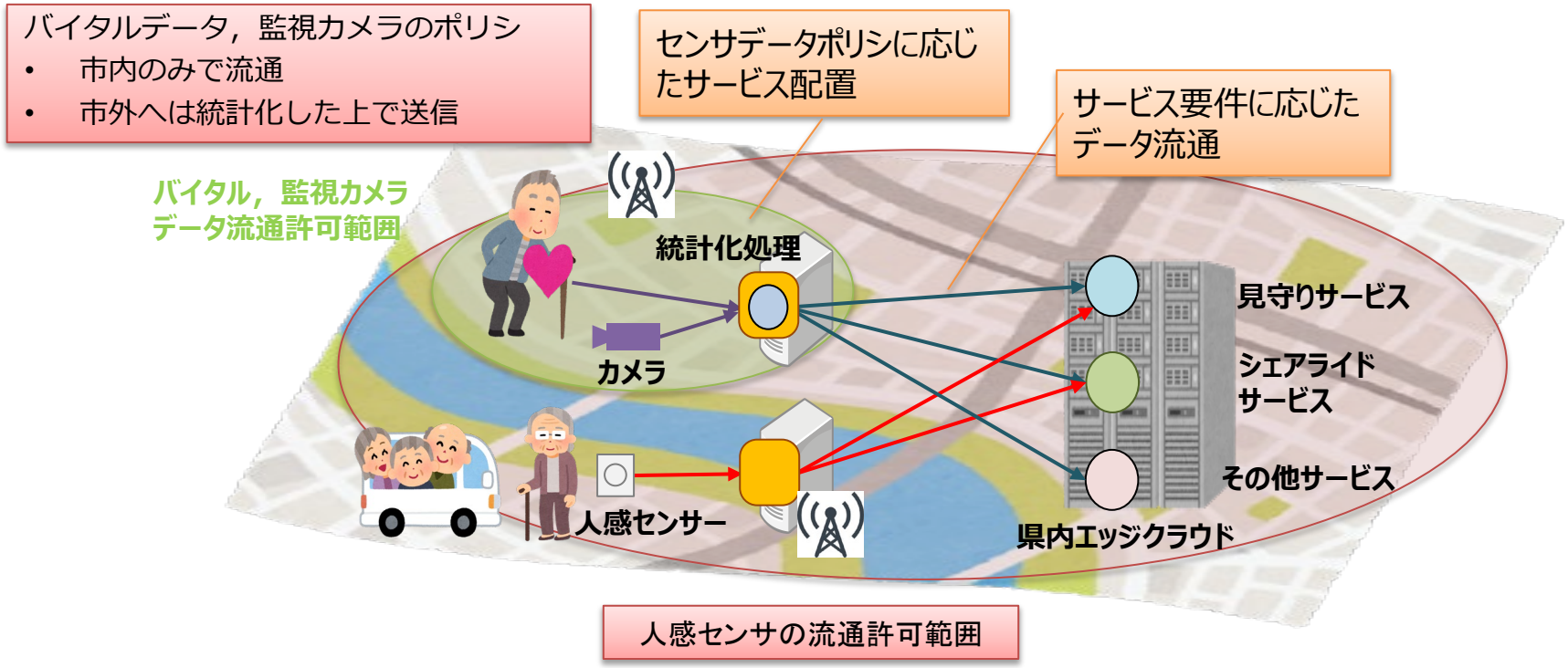


ユースケース１：ARの実験環境構築

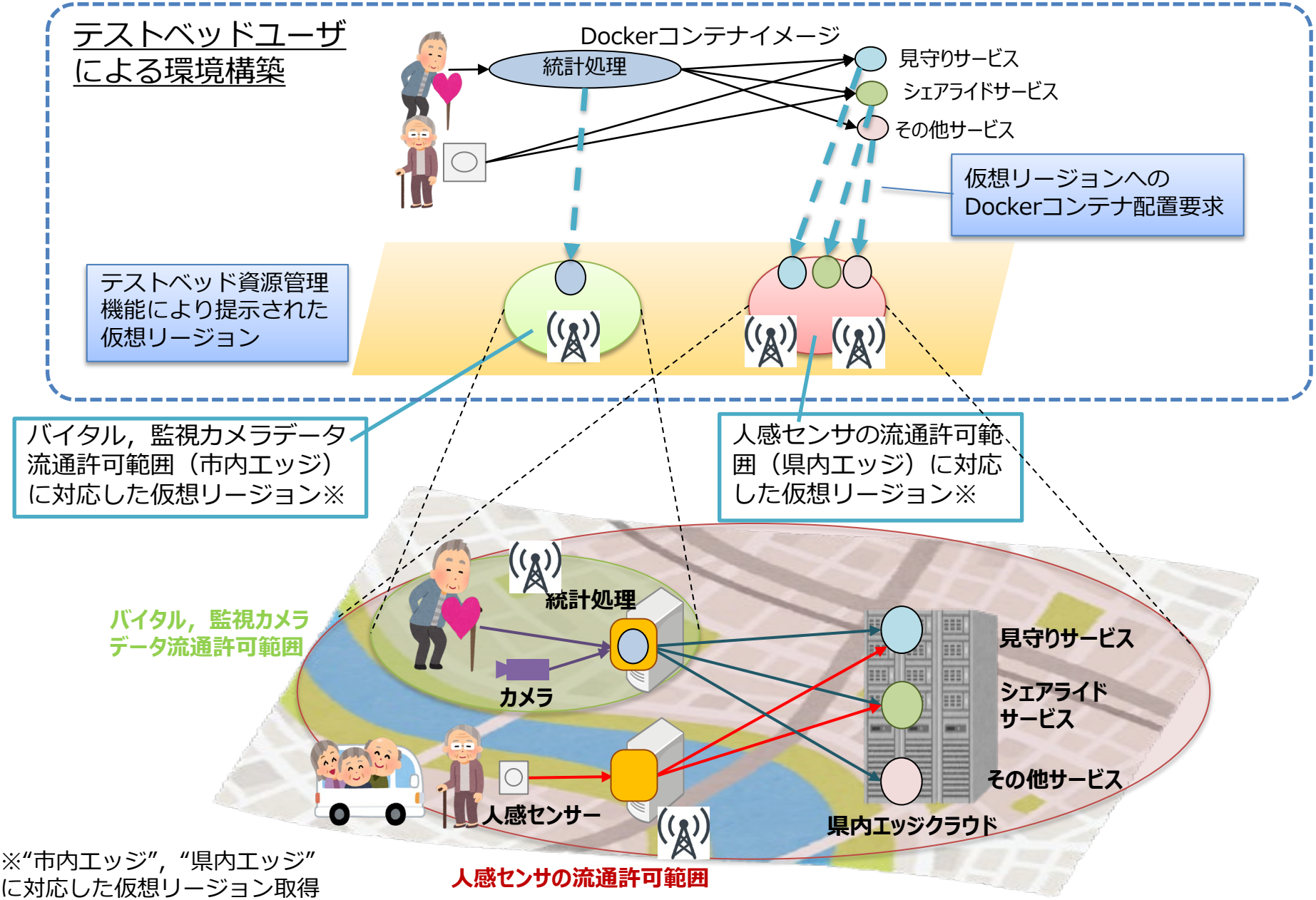


○プライバシポリシに基づくセンサデータの利用

◇プライバシポリシに基づく，センサデータの流通地域制限，データ内容の制限（送信先の制限，送信前のデータ加工等）



ユースケース２：ヘルスケア・見守りサービスの実験環境構築



※“市内エッジ”，“県内エッジ”に対応した仮想リージョン取得スクリプトを準備予定

テストベッドサービス展開状況

○StarBEDのサーバ，ネットワークを用いて構築

◇StarBEDネットワーク内での広域環境再現のため，ネットワーク上のリンクに伝搬遅延(Linux tc)を設定

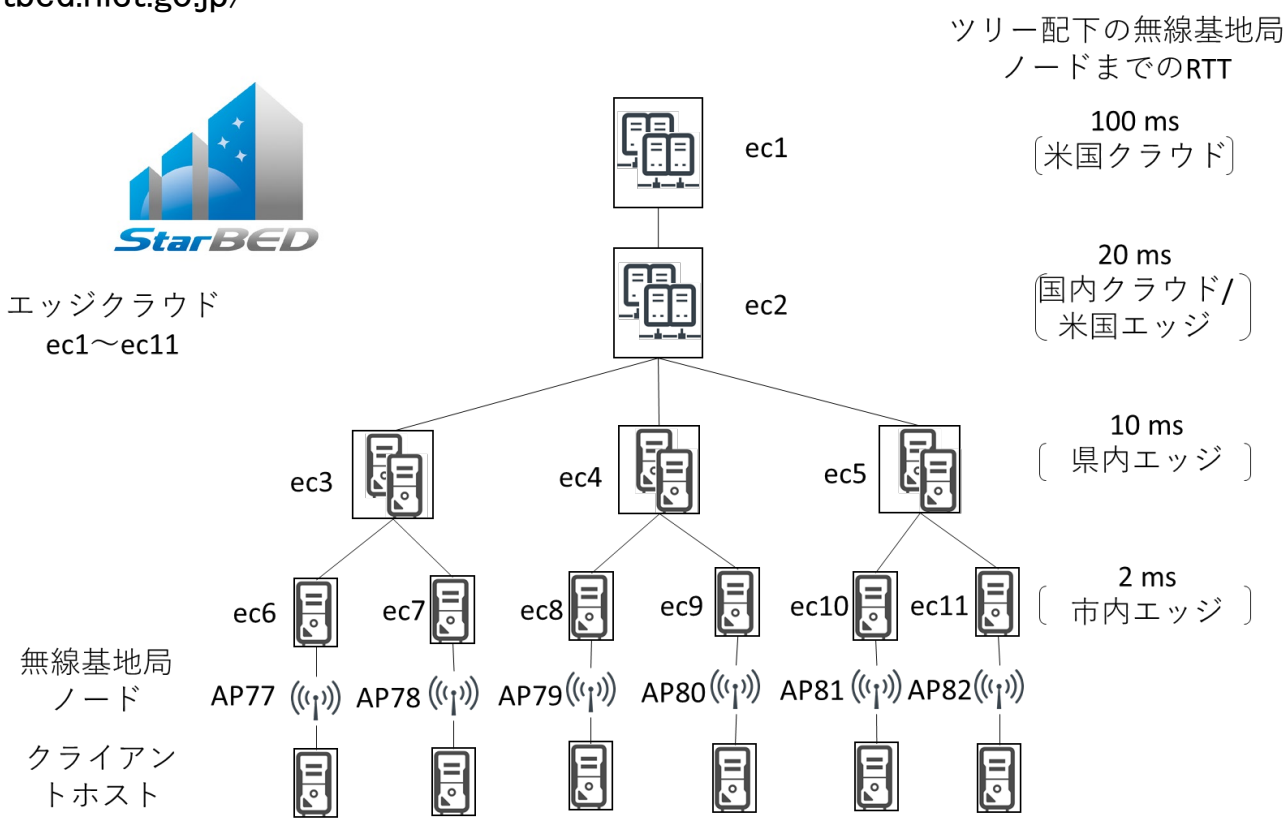
◇サーバスペック(計)

- CPU：164コア
- メモリ：724 GB
- HDD：2.7 TB

○NICT「総合テストベッド」HPにて利用者受付中

◇https://testbed.nict.go.jp/

エッジクラウド名	サーバ(VM)台数	性能(VM1台当たり)
ec1	2	CPU: 20コア，メモリ: 100GB，HDD: 500GB
ec2	2	CPU: 20コア，メモリ: 100GB，HDD: 500GB
ec3	1	CPU: 20コア，メモリ: 100GB，HDD: 500GB
ec4	1	CPU: 20コア，メモリ: 100GB，HDD: 500GB
ec5	1	CPU: 20コア，メモリ: 100GB，HDD: 500GB
ec6	1	CPU: 4コア，メモリ: 4GB，HDD: 32GB
ec7	1	CPU: 4コア，メモリ: 4GB，HDD: 32GB
ec8	1	CPU: 4コア，メモリ: 4GB，HDD: 32GB
ec9	1	CPU: 4コア，メモリ: 4GB，HDD: 32GB
ec10	1	CPU: 4コア，メモリ: 4GB，HDD: 32GB
ec11	1	CPU: 4コア，メモリ: 4GB，HDD: 32GB



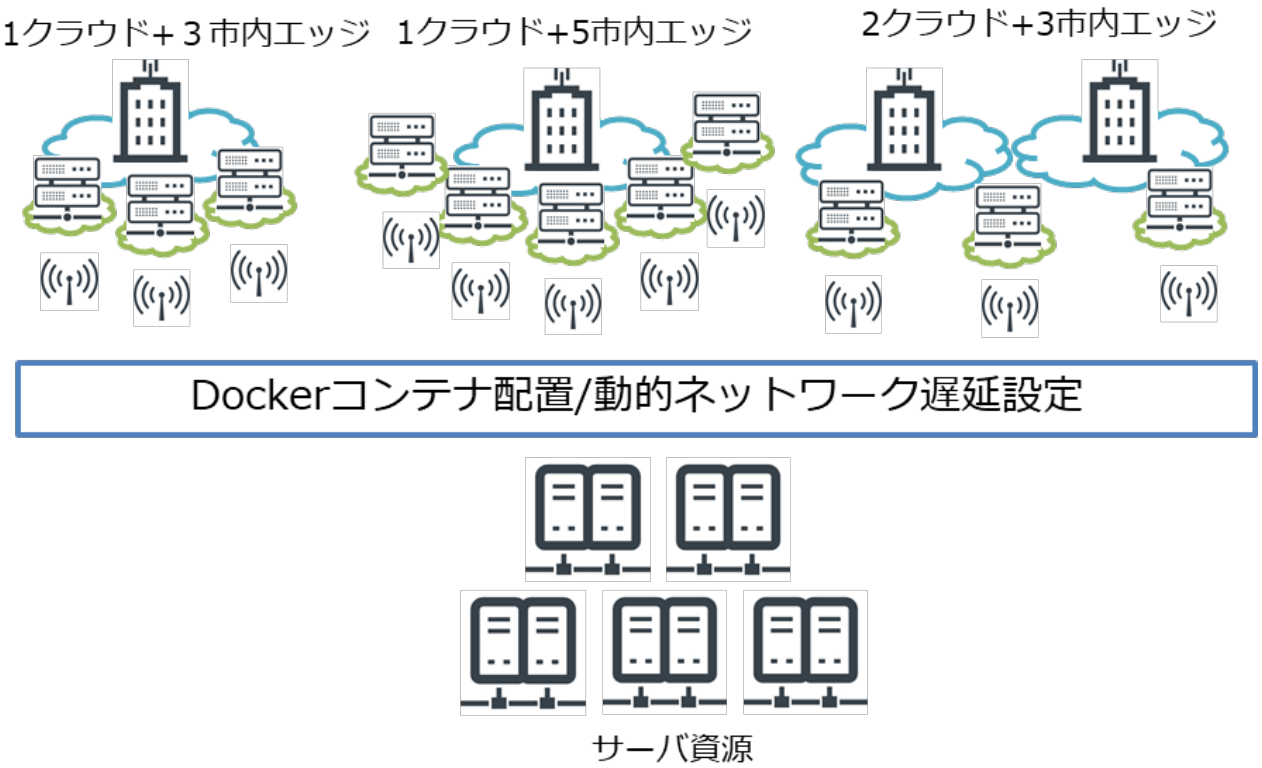
今後の展望

○ネットワーク遅延動的設定とDockerコンテナ配置の組合せによる資源配置

→テストベッドユーザ毎の実験ネットワーク規模・構成の自由度向上

○VXLANによる通信隔離

→テストベッドユーザの自由なIPアドレス利用



まとめ

○エッジコンピューティングテストベッドの展開

◇Dockerコンテナ管理機能（Kubernetes）をベースに，仮想リージョンの仕組みを実装，エッジコンピューティングテストベッドを構築

○ユースケース

◇応答遅延条件に応じたDockerコンテナ配置

◇地理的条件に応じたDockerコンテナ配置

○エッジコンピューティングテストベッドサービスとして展開中